Linear models for regression

$f: \mathbb{R}^d \longrightarrow \mathbb{R}$ the output will be a real value (regression)

$D = \{(x_n, t_n)_{n=1}^N\}$

t_n is now a real value

input-output pairs

① The simplest linear model for regression is a linear combination of the input variables

$$y(x; w) = w_0 + w_1 x_1 + \dots + w_d x_d = w^T x$$

parameters

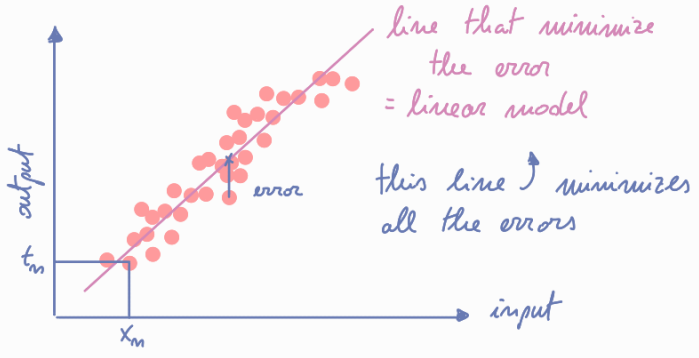
input

linear model of a linear function

$$X = [1, x_1, \dots, x_d]^T \; ; \; w = [w_0, w_1, \dots, w_d]^T$$

This is known as linear regression.
The key property is that it is a linear function of the parameters. It is also, however, a linear function of the input and this imposes significant limitations on the model.

The goal is to find an approximation of the true function but the model is linear so it will find an hyperplane that fits the distribution of the dataset



Considering the previous limitations, we extend the class of models by using non linear functions of the input variables.

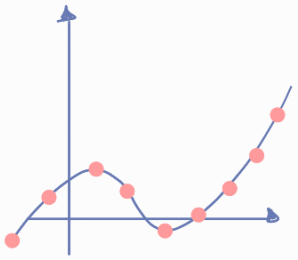
② Even more important than with classification is to consider basis functions and non linear transformations of the input.
We define a set of functions ϕ_i $i = 0, \dots, M$ (in total they are $M+1$) and consider such model:

$$y(x; w) = w^T \phi(x) \qquad \text{with } \phi(x) = \begin{bmatrix} \phi_0(x) \\ \vdots \\ \phi_M(x) \end{bmatrix} \quad \text{and } \phi_0(x) = 1$$

still M degrees of freedom

The resulting model is still linear in w but not in x (ϕ is usually non linear, otherwise this whole thing will be pointless).

e.g. we consider this particular transformation: $\phi = [x^0, x^1, \dots, x^M]^T$
$y(x; w) = w^T \phi(x) = (w_0, w_1, \dots, w_M)(x^0, x^1, \dots, x^M)^T = w_0 + w_1 x + w_2 x^2 + \dots + w_M x^M$
this is an M-degree polynomial



Solution of the problem is to find a good M such that we will best fit this data
← In this case we could chose $M = 3$

The choice of M is critical, it could lead to OVERFITTING: higher M could better fit the available data but we are interested in classifying new samples

There are several basis function → polynomial, radial, sigmoid, tanh

$\phi = [x^0, x^1, \dots, x^M]^T$

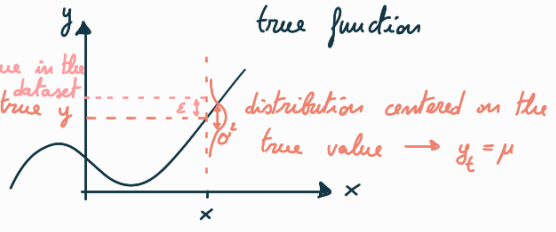
$\phi_i(x) = e^{-\frac{(x-\mu_i)^2}{2s^2}}$

$\phi_i(x) = \sigma\left(\frac{x-\mu_i}{s}\right)$

$\tanh(a) = 2\sigma(a) - 1$

$\sigma(a) = \frac{1}{1+e^{-a}}$

So far we just have seen an intuitive solution. What is the criterion to derive it?



Target values t in the dataset are given by the true function $y(x;w)$ affected by additive noise ε.

$$t = y(x;w) + \varepsilon$$

We assume that the noise follows a gaussian distribution.

We just described the same problem from a probabilistic point of view.

$$P(t \mid x, w, \beta) = \mathcal{N}(\varepsilon \mid y(x;w), \beta^{-1})$$

- $P(t \mid x, w, \beta)$ → likelihood
- $\mathcal{N}(\varepsilon \mid y(x;w), \beta^{-1})$ → the distribution is centered on the true value (zero-mean, we can translate it (?))
- variance is described in terms of precision β; precision is the inverse of the variance

Now we can try to maximize the likelihood (minimize the negative log likelihood)

maximize $P(\{t_1, \dots, t_N\} \mid x_1, \dots, x_N, w, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n \mid w^T \phi(x_n), \beta^{-1})$ ← under the hypothesis that the samples are independent and identically distributed we can use the theorem of total probability

or equivalently:

minimize $-\ln P(\{t_1, \dots, t_N\} \mid x_1, \dots, x_N, w, \beta) = \sum_{n=1}^{N} -\ln \mathcal{N}(t_n \mid w^T \phi(x_n), \beta^{-1})$

$$= \underbrace{-\beta \frac{1}{2} \sum_{n=1}^{N} (t_n - w^T \phi(x_n))^2}_{E_D(w)} + \underbrace{\frac{N}{2} \ln(2\pi\beta^{-1})}_{\text{do not depend on } w}$$

$E_D(w)$ is what we need to minimize

Maximum likelihood (minimum log likelihood) under zero-mean Gaussian assumption correspond to least square error minimization

$$\underset{w}{\text{argmax}}\ P(\{t_1, \dots, t_N\} \mid x_1, \dots, x_N, w, \beta) \longleftrightarrow \underset{w}{\text{argmin}}\ E_D(w) = \underset{w}{\text{argmin}}\ \frac{1}{2} \sum_{n=1}^{N} (t_n - w^T \phi(x_n))^2$$

sum of the squared errors: sum over all the points in the dataset of the squared difference between the true value t_n and the prediction $w^T \phi(x_n)$

linear regression can be solved with this optimization function that derives at the same time from the probabilistic formulation of the problem (maximum likelihood) and from the least square minimization (squared difference between the true value t_n and the prediction $w^T \phi(x_n)$).

From the closed form solution:

① compute the gradient $E_D(w) = \frac{1}{2}(t - \phi w)^T (t - \phi w)$ ← square

with $t = \begin{bmatrix} t_1 \\ \vdots \\ t_N \end{bmatrix}$ $(N \times 1)$, $\phi = \begin{bmatrix} \phi_0(x_1) & \phi_1(x_1) & \dots & \phi_M(x_N) \\ \phi_0(x_2) & \phi_1(x_2) & \dots & \phi_M(x_N) \\ \phi_0(x_N) & \phi_1(x_N) & \dots & \phi_M(x_N) \end{bmatrix}$ $(N \times M+1)$, $w = \begin{bmatrix} w_0 \\ \vdots \\ w_M \end{bmatrix}$ ← one weight for every element of the ϕ vector

② $\nabla E_D(w) = 0 \iff \phi^T \phi w = \phi^T t$

using M instead of $M+1$ for simplicity

$\phi^T \phi w = \phi^T t$

$(M \times N)(N \times M)(M \times \alpha) = (M \times N)(N \times 1)$

$(M \times M)(M \times \alpha) \qquad (M \times 1) \qquad \alpha = 1$

③ isolate w from $\phi^T \phi w = \phi^T t$

$$w = \boxed{(\phi^T \phi)^{-1} \phi^T} t$$

pseudo-inverse of $\phi \triangleq \phi^\dagger$

example

$f : \mathbb{R}^d \to \mathbb{R}$

$D = \{ (x_n, t_n)_{n=1}^N \}$

define t, w and ϕ

we assume this particular transformation:

$$\phi = \begin{matrix} 1 \\ \\ N \end{matrix} \begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ & \vdots & & \\ 1 & x_N & x_N^2 & x_N^3 \end{bmatrix}$$

$1 \to 0$, $3 \to M = 3$

the ϕ matrix will have N rows, one for each sample in the dataset and $M+1$ cols (number of features of the ϕ vector + ϕ_0

$$t = \begin{bmatrix} t_1 \\ \vdots \\ t_N \end{bmatrix} \qquad w = \begin{bmatrix} w_0 \\ \vdots \\ w_M \end{bmatrix}$$

our model will be:

$y(x; w) = w_0 + w_1 x + w_2 x^2 + w_3 x^3$

$w_{ML} = \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix} = \phi^\dagger t$ $\qquad$ $\text{pinv}(\phi) \cdot t$

w_{ML}: w from maximum likelihood

We need an iterative algorithm

Why do we need an iterative algorithm if we have a close form solution?
Because the ϕ matrix may be too large (N depends on the dataset size and M could also be very large)

Iterative algorithm : sequential learning

learning rate

$$\hat{w} \longleftarrow \hat{w} - \eta \nabla E_n$$

$$\hat{w} \longleftarrow \hat{w} - \eta (t_n - w^T \phi(x_n)) \phi(x_n)$$

this algorithm converges for suitable small values of η ; its value need to be chosen carefully

Regularization

Regularization is a technique to control over-fitting. We add a regularization term to an error function so that the whole error function to be minimized takes the form:

$$\text{argmin} \quad E_D(w) + \lambda E_W(w)$$

λ regularization coefficient

λ penalizes the weights that determine the overfitting that are too high in absolute value

A common choice for $E_W(w)$:

$$E_W(w) = \frac{1}{2} w^T w$$

this choice of regularizer is known as weight-decay because in sequential learning algorithms it encourages weight values to decay towards zero, unless supported by the data.

Another choice:

$$E_W(w) = \sum_{j=0}^{M} |w_j|^q$$

Putting all together :

$$\text{argmin} \quad \frac{1}{2} \sum_{n=1}^{N} (t_n - w^T \phi(x_n))^2 + \frac{\lambda}{2} w^T w$$

In general, when we add a parameter, we increase the flexibility of the model (better system that can more easily find a solution) but it needs to be well calibrated

Linear regression with regularization is called Ridge Regression

Everything can be extended to regression with multiple outputs

So far we have considered the case of a single target variable t. In some applications we may wish to predict $K>1$ target variables which we collectively denote with the target vector $\mathbf{t}$. This could be done by introducing a different set of basis functions for each component of $\mathbf{t}$, leading to multiple, independent regression problems; a more interesting and more common approach is to use the same set of basis functions to model all the components of the target vector.

$$f: \mathbb{R}^d \longrightarrow \mathbb{R}^K$$

$$\text{output } \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_K \end{bmatrix} = \begin{bmatrix} -\mathbf{w}_1^T- \\ -\mathbf{w}_2^T- \\ \vdots \\ -\mathbf{w}_K^T- \end{bmatrix} \begin{bmatrix} | \\ \boldsymbol{\phi} \\ | \end{bmatrix}$$

in each row we have the coefficients for a model

$$\mathbf{W}^T \quad (K \times M+1)$$

$\mathbf{W}$ is $(M+1 \times K)$
$\mathbf{W}^T$ is $(K \times M+1)$

This is still a linear model (linearity of the weight matrix).

The maximum likelihood is:

$$\ln P(\mathbf{T} \mid \mathbf{X}, \mathbf{W}, \beta) = \sum_{n=1}^{N} \ln \mathcal{N}(\mathbf{t}_n \mid \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1})$$

$$\mathbf{T} = \begin{bmatrix} \vdots \\ -\mathbf{t}_k- \\ \vdots \\ 1 \cdots K \end{bmatrix}$$

each row contains the output of the K models; it has one row for each sample in the dataset

Similarly as before we obtain:

$$\mathbf{W}_{ML} = \underbrace{(\boldsymbol{\phi}^T \boldsymbol{\phi})^{-1} \boldsymbol{\phi}^T}_{\text{pinv}(\boldsymbol{\phi})} \mathbf{T}$$

$\boldsymbol{\phi}^+$